CS 550-04 Operating Systems, Spring 2016
Programming Project 0

Out: Feb. 1 2016, Mon.
Due: Feb. 14 2016 Sun. 23:59:59

In this warm-up project, you are going to make some small changes to the xv6 code, then build and run xv6. The goal is for you to get acquainted with the xv6 system. Section 2 gives instructions on how to build and run the xv6 system. Section 3 and 4 describe the tasks of this project. Section 5 gives submission instructions. Section 6 clarifies how this project will be graded.

1 Baseline xv6 source code

For this and all the later projects, you will be working on the baseline xv6 code that needs to be downloaded from the course website (rather than cloned directly from the MIT repository). Your works will be submitted in the form of `diff` patches based on the xv6 baseline code (see details in Section 5).

The baseline xv6 source code for this project is here: `https://drive.google.com/open?id=0B5j-vGKxJzqxdV90ekZ1NDdrbVk`.

Note: Compiling and running xv6 source code downloaded elsewhere can cause consistent 100% CPU utilization on QEMU, which is not acceptable on the department's remote cluster. If you are working on the remote cluster, make sure to use the xv6 baseline code provided above. Points will be deducted if the system administrator reports that you ran QEMU with always 100% utilization (see details later in Section 6).

2 Build and run xv6

You can build and run xv6 using either a CS machine or your own computer.

2.1 Using a CS machine

(1) Log into a CS machine (i.e., a local machine or a remote cluster machine).
(2) Download the baseline xv6 source code, and do:

```
$ tar xvf cs550-16s-proj0-base.tar.gz
$ cp -r cs550-16s-proj0-base cs550-16s-proj0-working
```

Now you have two directories both containing the xv6 source code. *In the following description*, the directory "cs550-16s-proj0-base" is *notated* as `DIR_XV6_BASE`, and the directory "cs550-16s-proj0-working" is *notated* as `DIR_XV6_WORKING`.

The working directory (`DIR_XV6_WORKING`) is where you will hack, build and run the xv6 system.

The base directory (`DIR_XV6_BASE`) is used when generating the diff patch that you are going to submit.

(3) Download the Makefile patch from
`https://drive.google.com/open?id=0B5j-vGKxJzqxMi1RMUFrVXpzYUk` to
`DIR_XV6_WORKING`, and patch the working directory by running:

```
DIR_XV6_WORKING$ patch -p1 < xv6-cs550-Makefile.patch
```

(4) Compile and run xv6 without graphics

```
DIR_XV6_WORKING$ make qemu-nox
```

After the compiling the kernel source and generating the root file system, the Makefile would run the following command:

```
~zhangy/t_bin/qemu-system-i386 -nographic -hdb fs.img xv6.img -smp 2 -m 512
```

after which you would see the following output, indicating you've successfully compiled and run the xv6 system.

```
xv6...
cpu1: starting
cpu0: starting
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 ...
init: starting sh
$
```

If you prefer to use GDB to debug your work, you may also use the following to boot xv6:

```
DIR_XV6_WORKING$ make qemu-nox-gdb
```

Then open a new terminal session (if you are using the remote cluster, you need a new ssh session to the same remote machine), and run

```
DIR_XV6_WORKING$ gdb
```

If GDB is not connect to the listening port, run

```
DIR_XV6_WORKING$ target remote localhost:LPORT
```

where `LPORT` is the listening port you get from the other terminal session.

2.2 Using your own computer

You will need a Linux distribution that supports GNU C, which is targeted at the x86 architecture and using ELF binaries.

(1) Compile and install the MIT version of QEMU.

- Clone the QEMU repository:

  ```
  git clone https://github.com/geofft/qemu.git -b 6.828-1.7.0
  ```

- Change working directory to the root of the QEMU source tree, and configure it by:

  ```
  ./configure --disable-kvm
  ```

  The above command will install the binaries to the `/usr/local/bin` directory.

- Compile and install:

```
$ make
$ make install
```

"`make install`" may need root privilege.

(2) You may also need to install other tools or libraries. Suppose Ubuntu is used, you may need:

```
sudo apt-get isntall build-essential
sudo apt-get install git
sudo apt-get install zlib1g-dev
sudo apt-get install libglib2.0-dev
sudo apt-get isntall libpixman-1-dev
sudo apt-get isntall libfdt-dev
sudo apt-get install libtool
sudo apt-get install autoconf
...
```

(3) Perform Section 2.1 step (2).

(4) Perform Section 2.1 step (4).

Note: you don't need to patch the Makefile. The Makefile needs to be patched only if you are working on a CS machine.

3 (60 points) Add a new shell command (print messages in user space)

Add a new shell command (i.e., a user space program) named "proj0" to xv6. After this command is added, if doing an `ls` in the root directory, you will see:

```
$ ls
.              1 1 512
..             1 1 512
README         2 2 1973
cat            2 3 13636
echo           2 4 12601
forktest       2 5 8205
grep           2 6 15548
init           2 7 13502
kill           2 8 12721
ln             2 9 12627
ls             2 10 15483
mkdir          2 11 12750
rm             2 12 12739
sh             2 13 25383
stressfs       2 14 13721
usertests      2 15 67232
wc             2 16 14242
zombie         2 17 12355
proj0          2 18 12742
console        3 19 0
```

The second to the last line is the new program. The new command works like `echo`. But it prints a fix string ("CS550 proj0 print in user space: ") before the user-supplied string. For example, here should be the outputs if we run the program three times with different strings supplied:

```
$ proj0 Hello World
CS550 proj0 print in user space: Hello World
$ proj0 CS 550 Operating Systems Section 4
CS550 proj0 print in user space: CS 550 Operating Systems Section 4
$ proj0
CS550 proj0 print in user space:
```

Note: the system should operatable after boot (i.e., should not freeze).

4 (40 points) Print messages in kernel space

When compiling and booting the baseline xv6, the outputs during boot looks like:

```
xv6...
cpu1: starting
cpu0: starting
init: starting sh
$
```

In this task, add a message "CS550 proj0 printing in kernel space" just before "cpu0: starting" is printed. The desired output is:

```
xv6...
cpu1: starting
CS550 proj0 printing in kernel space: zhangy
cpu0: starting
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 ...
init: starting sh
$
```

Note: the system should operatable after boot (i.e., should not freeze).

5 Submit your work

- **Prepare the `diff` patch**: before generating the diff patch, remove all the compiled object files by running:

  ```
  make clean
  ```

  The diff patch file should be named "`proj0.your-bu-id.patch`". Run the following to generate the file:

  ```
  diff -uNr DIR_XV6_BASE DIR_XV6_WORKING >proj0.your-bu-id.patch
  ```

  where `DIR_XV6_BASE` is the path to the baseline xv6 code, and `DIR_XV6_WORKING` is the path to your xv6 working directory. If your BU email is "`abc@binghamton.edu`", the name of the patch should be "`proj0.abc.patch`".

- **Log your work**: besides the files needed to build your project, you must also include a README file which minimally contains your name and B-number. Additionally, it can contain the following:

  – The status of your implementation (especially, if not fully complete).

  - A description of how your code works, if that is not completely clear by reading the code (note that this should not be necessary, ideally your code should be self-documenting).
  - Possibly a log of test cases which work and which don't work.
  - Any other material you believe is relevant to the grading of your project.

- **Compress the files**: use the "`zip`" command in a Linux machine to compress the diff patch and your README file into a ZIP file. Name the ZIP file based on your BU email ID. For example, if your BU email is "abc@binghamton.edu", then the zip file should be named "proj0_abc.zip".

- **Submission**: submit the ZIP file to Blackboard before the deadline.

Suggestion: since the we will grade on a CS machine, test your code thoroughly (i.e., patch the baseline source using the to be submitted diff patch, compile, run, and test) on a CS machine before submitting.

6 Grading

The following are the general grading guidelines for this and all future projects.

(1) Compress your submission using the zip command in a Linux machine. If your zip file cannot be uncompressed, 5 points off.

(2) If the submitted ZIP file or the patch file inside is not named as specified above (so that it causes problems for TA's automated grading scripts), 10 points off.

(3) If the submitted diff patch contains object files, 10 points off.

(4) If you are to compile and run the xv6 system on the department's remote cluster, remember to use baseline xv6 source code provided by our course website. Compiling and running xv6 source code downloaded elsewhere can cause 100% CPU utilization on QEMU. If you are reported by the system administrator to be running QEMU with 100% CPU utilization on QEMU, 10 off the pending project.

(5) If the submitted patch cannot successfully patched to the baseline source code, or the patched code does not compile:

```
1   TA will try to fix the problem (for no more than 3 minutes);
2   if (problem solved)
3     1%-10% points off (based on how complex the fix is, TA's discretion);
4   else
5     TA may contact the student by email or schedule a demo to fix the problem;
6     if (problem solved)
7       11%-20% points off (based on how complex the fix is, TA's discretion);
8     else
9       All points off;
```

So in the case that TA contacts you to fix a problem, please respond to TA's email promptly or show up at the demo appointment on time; otherwise the line 9 above will be effective.

(6) If the code is not working as required in the project spec, the TA should take points based on the assigned full points of the task and the actual problem.

(7) Lastly but not the least, stick to the collaboration policy stated in the syllabus: you may discuss with you fellow students, but code should absolutely be kept private. Any kind of cheating will result in zero point on the project, and further reporting.